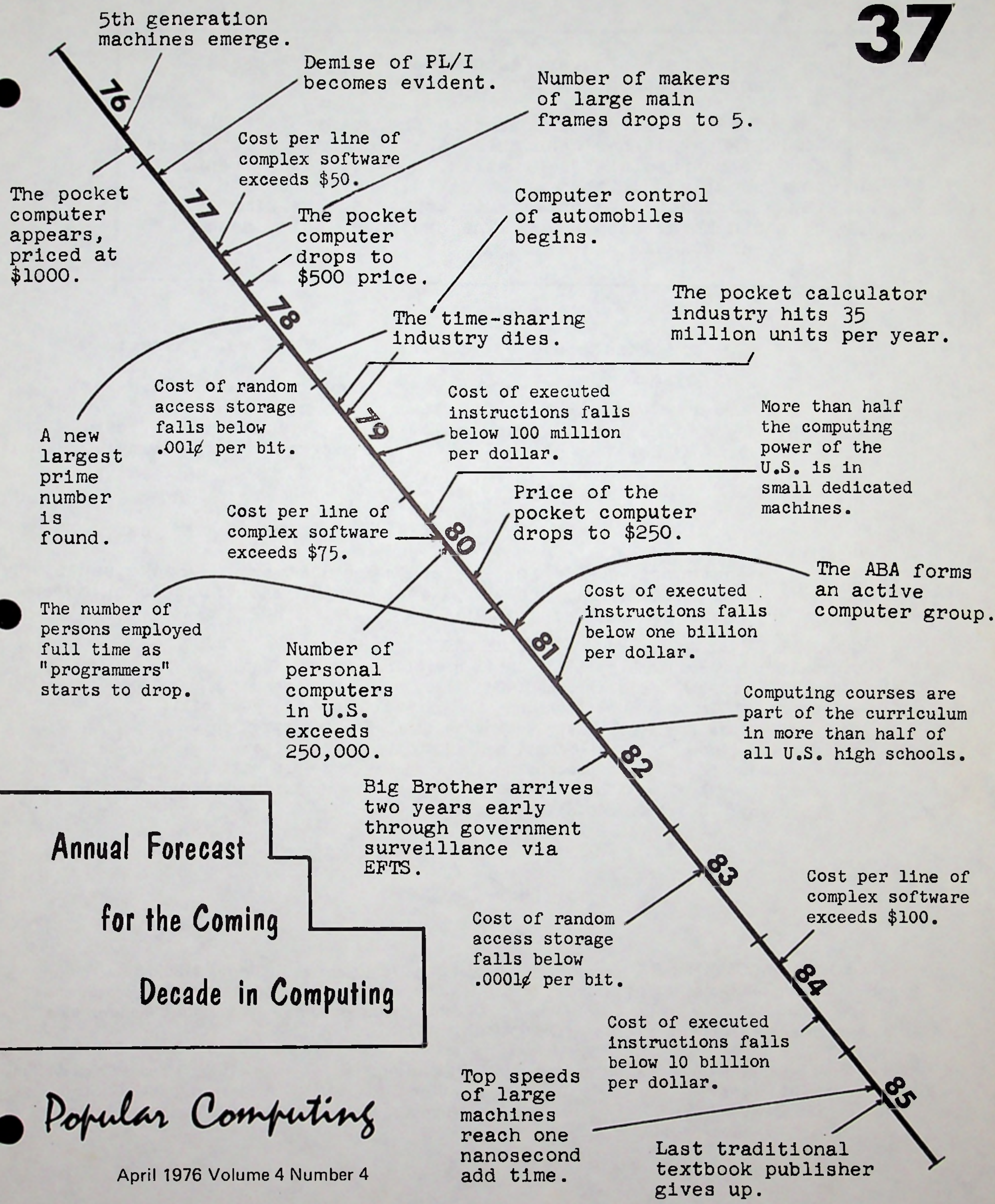


Annual Forecast for the Coming Decade in Computing

Popular Computing

April 1976 Volume 4 Number 4

This month's cover features our annual Forecast of events in computing for the coming decade.

The Forecast is prepared by the staff of POPULAR COMPUTING. It has been noted that long range forecasts tend to be pessimistic (that is, the predicted events come about sooner than anticipated) whereas short range forecasts are optimistic (that is, the time flies by and nothing happens). The crossover point seems to be around six years.

The price schedule as of June 1, 1976 is as follows:

All prices are for remittance in U.S. dollars with order. Add $3 to all prices if we must bill.

The basic subscription price is $16 for one year, or $30 for two years. Add for postage and handling:

U.S. $1.50 per year
Canada and Mexico $3 per year
All other countries $5 per year

Above prices are for surface mail delivery. Back issues, when available, are $2.50 each, or $2 each for two or more (same or different issues).

THERE ARE NO DISCOUNTS FROM THE ABOVE PRICES.
Overseas orders cannot be filled until remittance is received.
Non-printed purchase orders cannot be filled until remittance is received.
We cannot handle open subscriptions; the renewal notice is your bill.
Renewal notices are sent to the addressee of the subscription only.
Third party (i.e., agency) orders, add $3 to the above prices.
Agency orders requiring reprocessing are subject to a $3 penalty.
Copies not received will be replaced free if we are notified within 90 days (U.S.) or 120 days (overseas).

POPULAR COMPUTING is published monthly at Box 272, Calabasas, California 91302. Subscription rate in the United States is $18 per year, or $15 if remittance accompanies the order. For Canada and Mexico, add $4 per year to the above rates. For all other countries, add $6 per year to the above rates. Back issues $2 each.

Stack Action

A main routine outputs a series of numbers that are integers in the range from 001 to 999; they are produced randomly.

A subroutine is to place these numbers in a stack (such as a block of consecutive words of storage) in ascending order, with the number 1000 at the end of the stack as a terminator. The process will end when there are 50 numbers in the stack.

There is one restriction: no number shall be added to the stack if its value is within 10 of any previously stacked number. For example, if the number 383 has been placed in the stack, then any number in the ranges 373--382 or 384--393 (and, of course, a repeat of 383) is to be rejected (i.e., by-passed).

For example, for the sequence produced in the left column, the numbers on the right show the successive contents of a block of storage addressed at T, T+1, T+2, etc.

start		1000						
383		383	1000					
572		383	572	1000				
911		383	572	911	1000			
100		100	383	572	911	1000		
582	reject							
250		100	250	383	572	911	1000	
908	reject							
110	reject							
398		100	250	383	398	572	911	1000
		T	T+1	T+2	T+3	T+4	T+5	T+6

What is the logic of the action called for by the subroutine?

Outline a procedure to test that logic.

Contest No. 6

In the article "Reading, Writing, and Computing" in the November-December issue of <u>Creative Computing</u>, Walter Koetke presents this problem: Is each positive integer 1 through N a divisor of some integer that contains only the digits one and zero?

The answer is yes, as shown by David Ferguson:

Any number N, not divisible by 2 or 5, divides $10^{\phi(N)} - 1$. If $N = 2^a 5^b m$, then N divides $(10^{\phi(m)} - 1) \cdot 10^{\max(a,b)} = M(N)$. $\phi(x)$ is the Euler function: the number of numbers less than x relatively prime to x.

A list of such numbers, for the integers from 1 to 40, is given in the accompanying table. Any other digit could be substituted for the digit <u>one</u>.

A possible algorithm for finding such numbers is given, using N = 29 as an example. We start with the four possible leading three digits (100, 101, 110, 111) and record the remainders on division by 29. Each of these remainders can be extended by appending a zero or a one, leading to a new set of remainders. The process continues as shown until a remainder of zero is found, after which the sequence leading to that zero (shown by the arrows) is readily traced. There is no assurance that this algorithm will give the smallest possible result.

The suggested algorithm is probably not efficient, and would be awkward to implement for large values of N, since the list of remainders to be kept track of is of the order of N itself.

For our 6th contest, the usual $25 prize goes to the person submitting the longest list (produced by computer, of course) like that of Table A. The list must be of consecutive integers, and must include the other factor, as shown. A flowchart (or equivalent) of the logic used to derive the results must be included. All entries must be received by POPULAR COMPUTING by June 30, 1976.

100	13	50	21
101	14	51	22
110	23	60	2
111	24	61	3
130	14	180	6
131	15	181	7
140	24	190	16
141	25	191	17
230	27	200	26
231	28	201	27
240	8	220	17
241	9	221	18
140	24	30	1
141	25	31	2
150	5	40	11
151	6	41	12
250	18	210	16
251	19	211	17
270	9	70	12
271	10	71	13
280	19	160	15
281	20	161	16
80	22	170	25
81	23	171	26
90	3	260	28
91	4	261	0

$$29 \overline{)1101101}\quad = 37969$$

A possible algorithm for deriving the number composed of only ones and zeros that is divisible by 29.

N	times	product
1	1	1
2	5	10
3	37	110
4	25	100
5	2	10
6	185	1110
7	1443	10101
8	125	1000
9	12345679	111111111
10	1	10
11	1	11
12	925	11100
13	77	1001
14	715	10010
15	74	1110
16	625	10000
17	653	11101
18	61728395	1111111110
19	5269	100111
20	5	100
21	481	10101
22	5	110
23	43957	1011011
24	4625	111000
25	4	100
26	385	10010
27	411522593	11111110011
28	3575	100100
29	37969	1101101
30	37	1110
31	322581	10000011
32	3125	100000
33	336397	11101101
34	3265	111010
35	286	10010
36	280558641975	10100111111100
37	3	111
38	2895	110010
39	259	10101
40	25	1000

Table A. The first 40 integers, with numbers they divide, these numbers made up solely of 1's and 0's.

Problem Solution

In essay No. 8 in the Art of Computing series (PC28-3) the problem posed was this:

> A man cashes a check at a bank. The teller mistakenly interchanges the amount for dollars and the amount for cents, so that the man receives more than the amount of the check. The man finds that, after spending \$14.19, he still has twice as much as the check called for. What was the amount on the check?

In attempting to solve the problem algebraically, we were led to the equation

$$199D - 98C + 1419 = 0 \qquad \text{(A)}$$

Since this single equation has two variables, ordinary methods of algebra will not do for solving it. If we solve the equation for C, we have

$$C = \frac{199}{98} D + \frac{1419}{98}$$

which can be written (separating out whole numbers):

$$C = 2D + \frac{3}{98} D + 14 + \frac{47}{98} \qquad \text{(B)}$$

The second condition on equation A (and all subsequent equations) is that each variable can take on only integral values. In equation B, then, the indicated terms are intrinsically integral, and hence the other two terms must also represent integers. This fact can be represented as:

$$\frac{3}{98} D + \frac{47}{98} = E$$

or

$$3D + 47 = 98E$$

and, solving for D:

$$D = \frac{98}{3} E - \frac{47}{3}$$

Separating out the known integers as before:

$$D = 32E + \frac{2}{3} E - 15 - \frac{2}{3}$$

and this process can be repeated:

$$\frac{2}{3} E - \frac{2}{3} = F$$

$$2E - 2 = 3F$$

$$E = \frac{3}{2} F + 1$$

until we reach a stage at which suitable integer values can be found. In this case, F = 0 will do nicely and, substituting back, we find that D = 17 and C = 49.

The process just described is awkward to apply to the general case with literal coefficients, as in the third problem presented in issue No. 28:

3 A man cashes a check at a bank. The teller mistakenly interchanges the amount for dollars and the amount for cents, so that the man receives more than the amount of the check. The man finds that, after spending X dollars, he still has twice as much as the check called for. What are the values of X that make this a real problem?

However, we can go over the specific case and gain some insights into a general solution. The following material comes from Jules Mersel.

Equation (A) may be expressed as follows:

$$199D + 1419 \equiv 0 \bmod 98$$
$$3D + 1419 \equiv 0 \bmod 98$$

The first insight comes from noting that 1419 is a cumbersome number which may be reduced to its lowest possible value modulo 98:

$$3D + 14(98) + 47 \equiv 0 \bmod 98$$
$$3D + 47 \equiv 0 \bmod 98$$

Similarly, since 47 is not a multiple of 3, we may force the relation:

$$3D - 51 \equiv 0 \bmod 98$$
$$D \equiv 17 \bmod 98$$

and, since $0 \leq D \leq 99$, D = 17.

Working backwards now, with the value 17 for D, we find that C = 49 and the original check (as before) was $17.49.

In general, then, we have

$$100C + D - X = 2(100D + C) \qquad (1)$$

$$199D + X = 98C \qquad (2)$$

$$199D + X \equiv 0 \bmod 98 \qquad (3)$$

$$3D + X \equiv 0 \bmod 98 \qquad (4)$$

and our insight gained from the special case lets us write for X:

$$X = 98m + r \quad \text{where } 0 \leq r \leq 97$$

In turn, $$r = 3n + p \quad \text{where } 0 \leq n \leq 32$$

and $$p = 0, 1, \text{ or } 2$$

or $$X = 98m + 3n + p \qquad (5)$$
$$\text{where } 0 \leq m \leq 102$$

$$3D + 98m + 3n + p \equiv 0 \bmod 98$$

$$3D + 3n + p \equiv 0 \bmod 98 \qquad (6)$$

and there are three cases:

Case I: $p = 1$
Case II: $p = 2$
Case III: $p = 0$

Case I ($p = 1$):

$$3D + 3n + 1 \equiv 0 \bmod 98$$

$$3D + 3n + 99 \equiv 0 \bmod 98$$

$$D \equiv -n - 33 \bmod 98$$

$$D \equiv 65 - n \bmod 98$$

and since $0 \leq D \leq 99$ and $0 \leq n \leq 32$

$$D = 65 - n$$

Applying this value to equation (2):

$$98C = 199(65 - n) + x$$
$$= 199(65 - n) + 98m + 3n + 1$$
$$= 98m - 196n + 12936$$
$$= 98m - 196n + 98(132)$$

but $0 \leq C \leq 99$ therefore $0 \leq m - 2n + 132 \leq 99$
or $132 \geq 2n - m \geq 33$

since $n \leq 32$ and $m \geq 0$,

a solution for Case I exists only when

$$2n - m \geq 33$$

and then $$D = 65 - n$$

$$C = 132 + m - 2n$$

For example, choose n = 20, m = 4:

$$x = 4(98) + 3(20) + 1 = 453$$

$$D = 65 - 20 = 45$$

$$C = 132 + 4 - 40 = 96$$

and we have 96.45 - 4.53 = 91.92 = 2(45.96)

Case II (p = 2):

From equation (6):

$$3D + 3n + 2 \equiv 0 \bmod 98$$

$$3D + 3n - 96 \equiv 0 \bmod 98$$

$$D + n - 32 \equiv 0 \bmod 98$$

$$D \equiv 32 - n \bmod 98$$

since $0 \leq D \leq 99$ and $n \geq 0$,

$$D = 32 - n$$

and, applying this value to equation (2):

$$\begin{aligned} 98C &= 199D + x \\ &= 199(32 - n) + 98m + 3n + 2 \\ &= 98m - 196n + 6370 \\ C &= m - 2n + 65 \end{aligned}$$

but $0 \leq C \leq 99$ therefore $0 \leq m - 2n + 65 \leq 99$
or $-65 \leq m - 2n \leq 34$

since $n \leq 32$, there is a solution to Case II only when

$$2n - m \geq 34$$

$$D = 32 - n$$

$$C = 65 + m - 2n$$

For example, choose n = 30 and m = 20:

$$x = 20(98) + 30(3) + 2 = 2052$$

$$D = 32 - 30 = 2$$

$$C = 65 + 20 - 2(30) = 25$$

and 25.02 - 20.52 = 4.50 = 2(2.25)

Case III ($p = 0$):

From equation (6): $3D + 3n \equiv 0 \mod 98$

$$D \equiv -n \mod 98 \quad \text{since } 0 \leq D \leq 99$$

III(a) $D = 0$ if $n = 0$
III(b) $D = 98 - n$ if $n > 0$

Applying $D = 0$ to equation (2):

$$98C = 199(0) + x$$
$$= 199(0) + 98m$$
$$C = m$$

So a result exists for Case III(a) when $m \leq 99$.

Applying $D = 98 - n$ to equation (2):

$$98C = 199(98 - n) + 98m + 3n$$
$$= 98m - 196n + 98(199)$$
$$C = m - 2n + 199$$

since $0 \leq C \leq 99$, therefore $0 \leq m - 2n + 199 \leq 99$

but, since $1 \leq n \leq 32$ and $0 \leq m \leq 102$, there is no solution to Case III(b).

So for Case III ($p = 0$), a solution exists only for

$$n = 0 \text{ and } m \leq 99$$

and then

$$D = 0$$
$$C = m$$

As an example, choose $m = 20$:

$$x = 98(20) + 0 + 0 = 1960$$
$$D = 0$$
$$C = 20$$

and $20.00 - 19.60 = 0.40 = 2(0.20)$

Jared Weinberger, Bologna, Italy, points out that the sequence given as an illustration for SLOWGROW (PC30-10) is in error. The sequence starts with any integer and forms each subsequent term by adding to the preceding term the sum of its digits. Thus, the example given should have begun

7 14 19 29 40 44 52 59 73 ...

Log 37	1.568201724066994996808450689539129447982972690166313
ln 37	3.610917912644224444368095671031447163900077587167636
$\sqrt{37}$	6.082762530298219688999684245202067062084970094786411
$\sqrt[3]{37}$	3.332221851645953260095450505185119004409616671950062
$\sqrt[10]{37}$	1.434895165675359452829405842182258706024214075867060
$\sqrt[100]{37}$	1.036769033849400879875881175233017497034970001106672
e^{37}	11719142372802611.30877293979119019452167536369446182
	23834809114359096826486861321089074
π^{37}	2480534602660760780.433710790045522100100824031419549
	833903269349676966827477951312798
$\tan^{-1}$ 37	1.543775877607631830443146358281231809903733691908670

N-SERIES 37

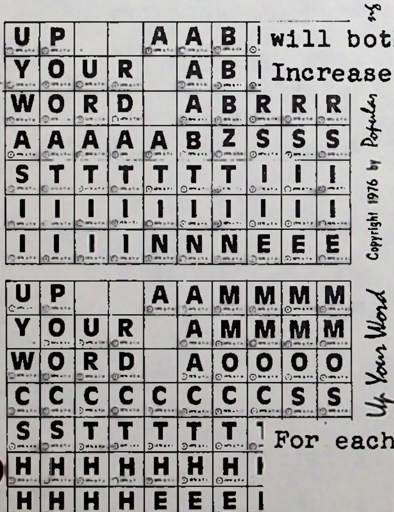
M	O	N	O	P	O	L			?
	C	R	A	B	B	L	E		?
M	A	S	T	E	R	M	I		?

...balderdas !

IT'S TIME TO TRY *Up Your Word*

THE NEWEST, MOST CHALLENGING WORD GAME

For two or more players.



will both challenge and entertain you and your friends.
Increase your word skill--organize a tournament!

The complete game, along with instructions and several variations, is only $2.00!

ORDER FROM *Popular Computing*

BOX 272 CALABASAS, CA. 91302

For each game, send $2 plus 30¢ for postage and handling (U.S., Canada, and Mexico)
or plus 50¢ for postage and handling (all other countries)

For two games, send $4 plus 50¢ (U.S., Canada, and Mexico)
or plus 75¢ (all other countries)

California residents add 12¢ per game.

Life or Death

The following game is to be played. A number is entered in one cell of an infinite grid. At each stage of the game, the number in a cell dictates a move in one of 8 directions:

1	2	3
4	X	5
6	7	0

according to X modulo 8. For example, if the number in cell X is 5, or 13, or 21, or 69, the move is made to the east. To make the move, the value of X is reduced by one, and the cell it points to will contain either X/2 (if X was even) or 3X+1 (if X was odd). No further movement arises from a cell containing the value one.

All moves for a stage are made simultaneously. When two or more cells are all creating new numbers for the same cell, the lowest number prevails.

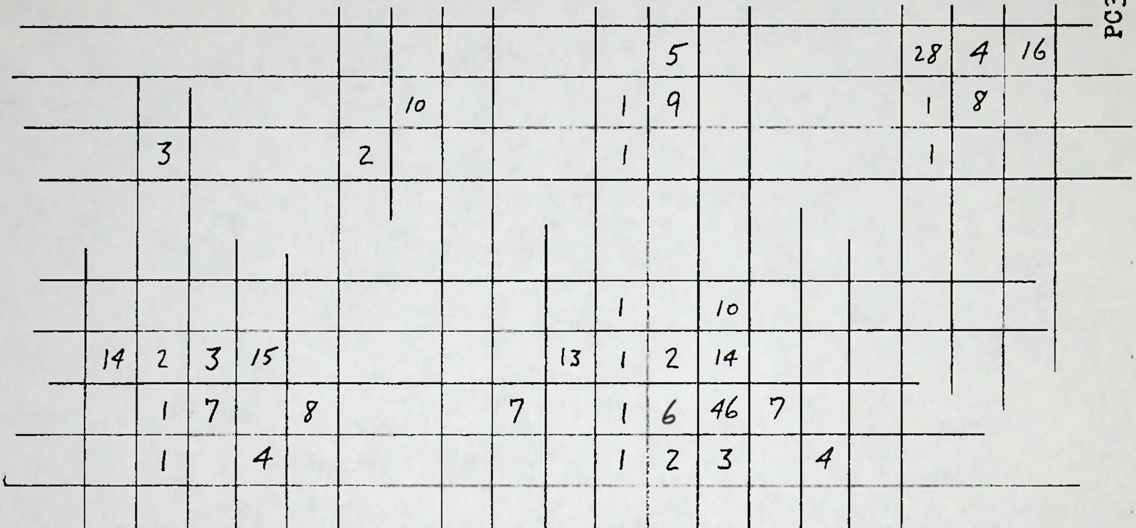
If the game begins with the number one, it is immediately over. If the game begins with 2, it is over in one move. Things get interesting when the game begins with 3. The accompanying figure shows the first ten stages for the game beginning with 3. It can be seen that large blocks of ones develop, since it is well known that the 3X+1 process always converges to one (see the discussions of the 3X+1 process in issues 1, 4, 13, and 25).

We have then these problems:

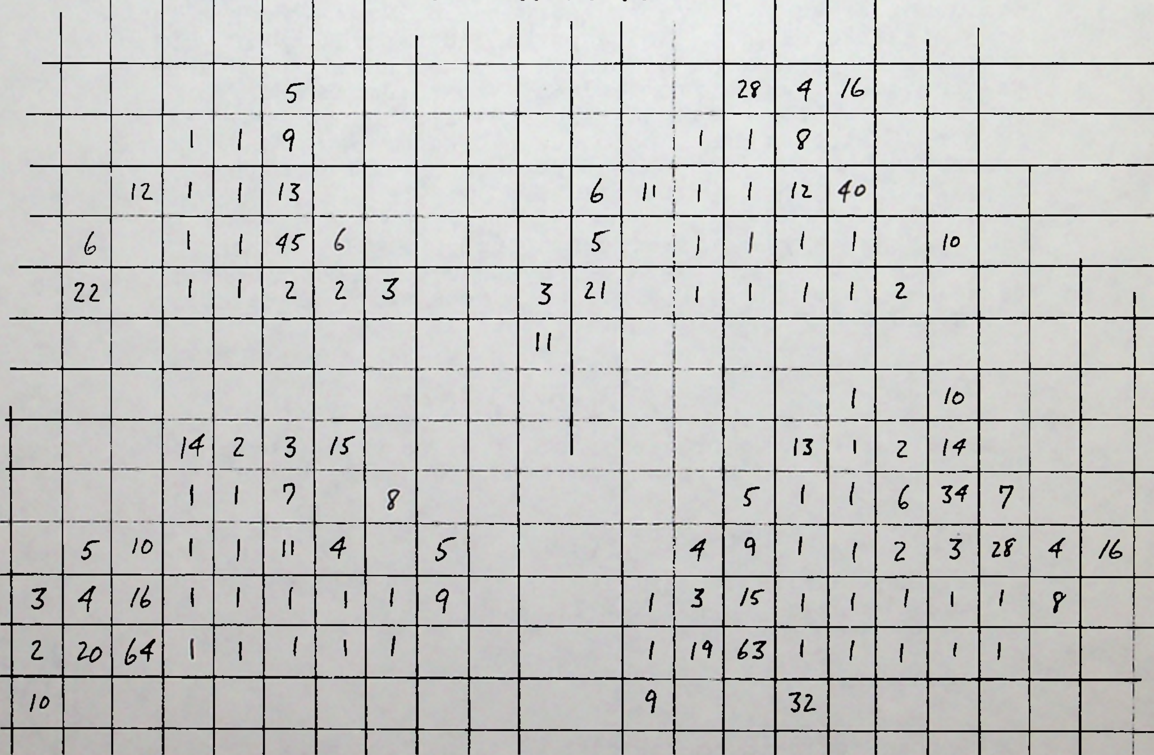
(1) Write a program to carry out the Life or Death procedure from stage to stage.

(2) Will any pattern stabilize; that is, will any pattern converge to all ones?

(3) What is the developing pattern for starting values other than 3?



Life or Death



Problem Solution

The IBM Card Problem

NUMBER 55

Problem 55 called for:

A. The logic of a subroutine that furnishes the number of paths from hole (1,9) to hole (column X, row Y) of a standard IBM card.
B. A Fortran program for that logic.
C. The number of paths from (1,9) to (80,12).

Contributing editor Thomas R. Parkin writes:

Let us discuss this problem in terms of the vertical and horizontal segments of the paths from one point to another. All paths must proceed from left to right and from bottom to top, only. Thus we may assign 0 to each vertical segment from one row to another and 1 to each horizontal segment from one column to another. Thus to go from (1,9) to (3,8), we can go across 2 columns and up 1 row, or across 1 column and up 1 row and across 1 column, or go up 1 row and across 2 columns--for a total of 3 possible paths. These paths correspond to 110, 101, and 011, respectively, according to our labelling. Thus we see that if we go from column A to column B and from row C to row D, we will have (B - A) 1's and (D - C) 0's in our final path descriptions. (Of course, we have converted the designations of rows on an IBM card from 12, 11, 0, 1, 2,...,9 into 12, 11, 10, 9,...,1).

Let us say we traverse N columns (N = B - A) and K rows (K = D - C). Then we have exactly as many paths as we can form binary numbers each consisting of exactly N 1's and K 0's and, of course, this is

$$P_{(N,K)} = C^{N}_{N+K} = \frac{(N+K)!}{N!\ K!}$$

Thus, if we wish to proceed from (1,9) to (3,8), we are going up 1 row and across 2 columns, so N = 2, K = 1 and

$$P_{(N,K)} = P_{(2,1)} = 3!/1!2! = 3$$

as we found earlier. Similarly, to go from (1,9) to (80,12), we have N = 79, K = 11, and

$$P_{(79,11)} = 90!/79!11! =$$

$$\frac{90\cdot 89\cdot 88\cdot 87\cdot 86\cdot 85\cdot 84\cdot 83\cdot 82\cdot 81\cdot 80}{1\cdot 2\cdot 3\cdot 4\cdot 5\cdot 6\cdot 7\cdot 8\cdot 9\cdot 10\cdot 11} = 41604694413840$$

The pertinent Fortran program follows.

```
      SUBRØUTINE PATHS (IC, IR, IP, AP, ILM)
C          This subroutine calculates the number of paths
C          from hole to hole in an 80 x 12 hole card,
C          from bottom row, left column (or IC = 1,
C          IR = 9) to the hole at (IC, IR), expressed
C          in standard IBM card numbering conventions.
C          The paths are constrained to go from left to
C          right and bottom to top only.   Columns are
C          numbered 1 to 80 left to right, rows are
C          numbered 12, 11, 0, 1, 2,...,9 from top to
C          bottom.
C
C          Calling sequence:  CALL PATHS (IC, IR, IP, AP, ILM)
C          where IC is the column number and
C                IR is the row number of the terminating
C                point.
C          IP will contain the number of paths between hole
C          (1,9) and hole (IC,IR) upon return, provided
C          the integer limit of the particular Fortran
C          being used can accommodate the number; if not,
C          IP = -1, and the floating point value of the
C          number of paths will be in AP.
C          ILM is preset to the (Fortran integer limit -1).
C
C          The following statements are inserted to skip
C          the calculations if parameters outside the
C          proper limits are given.
C
      IF (IR.EQ.10) GØ TØ 50
      IF (IR.LT.00) GØ TØ 50
      IF (IR.GT.12) GØ TØ 50
      IF (IC.LT.00) GØ TØ 50
      IF (IC.GT.80) GØ TØ 50
C
C          These could be combined into one compound
C          logical statement if the particular Fortran
C          will allow this form.
C
C          We will now convert the input parameters to
C          internally useful ranges.   We shall let K run
C          from 0 to 11 corresponding to inputs of rows of
C          9,8,...,0,11,12, and N run from 0 to 79
C          corresponding to columns of 1 to 80.
C
      K = 9 - IR
      IF (IR.GE.11) K = IR - 1
      N = IC - 1
C          Now check for special cases of N or K or both
C          being zero.
C
      IF (N.EQ.0) GØ TØ 30
      IF (K.EQ.0) GØ TØ 40
C
C          If we fall through to here, neither N nor K is
C          zero, and we can proceed.   In order to calculate
C          the number of paths, we must evaluate the
C          formula P(N,K) = (N+K) FACTORIAL DIVIDED BY
C          N FACTORIAL TIMES K FACTORIAL.
C
```

PC37-18

```
C         In order to shorten the calculation as much as
C         possible, we will first find out which is larger,
C         K or N, and cancel off that many factors from
C         both the numerator and denominator.
C         To do this, we will preset the limit of the
C         DØ LØØP index to one of the variables and
C         reverse this if appropriate.
C
      IX = N
      IF (N.GT.K) IX = K
C
C         The DØ LØØP limit is now the smaller of N or K.
C
C         Now perform the evaluation of the factorial
C         formula.   Note that we have the same number of
C         factors in both the numerator and denominator
C         after cancelling the low order terms of the
C         numerator against either N or K (whichever is
C         larger) terms of the denominator.  Furthermore,
C         we shall work in floating point (we assume the
C         floating point limit is greater than 4.2*10**12).
C         We shall use a floating value of K or N
C         (whichever is smaller) and our floating point
C         answer space.
C
      IY = K
      IF (N.GT.K) IY = N
      AY = FLØAT (IY)
      AI = 0.0
      AP = 1.0
      DØ 10 I = 1, IX
      AI = AI + 1.0
      AP = AP*(AY + AI)/AI
   10 CØNTINUE
C
C         Our answer is now in AP.   We must check the
C         integer limit and tidy up the special cases
C         of N or K or both equal to zero.
C
      B = FLØAT (ILM)
      IF (AP.GT.B) GØ TØ 20
      IP = IFIX (AP)
      RETURN
   20 CØNTINUE
      IP = -1
      RETURN
   30 CØNTINUE
      IF (K.EQ.0) GØ TØ 50
   40 CØNTINUE
      IP = 1
      RETURN
   50 CØNTINUE
      IP = 0
      RETURN
      END
```